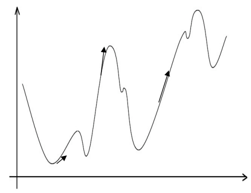
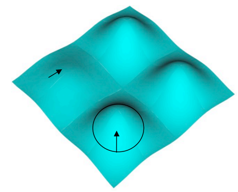
All About Morse-Smale Complexes

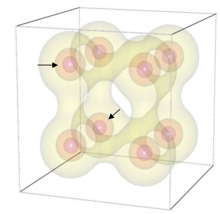
What is a Morse-Smale Complex?
Simplification: cancellations in the complex.
An algorithm to compute for 2D PL data.
A discrete algorithm for 3D+ data.

Gradient Vectors

- The gradient of a differentiable scalar function *f* at a point *p* is $\nabla f(p) = (\frac{\partial f}{\partial x_0}(p), \ldots, \frac{\partial f}{\partial x_n}(p))$
- Direction of steepest ascent
  - perpendicular to contours



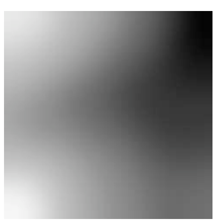


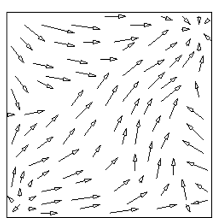


Gradient Vector Field

- The vector field given by the gradient vectors of a scalar function is the *gradient vector field*

$$\nabla f = (\frac{\partial f}{\partial x_0}, \ldots, \frac{\partial f}{\partial x_n})$$

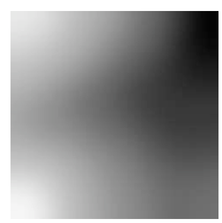


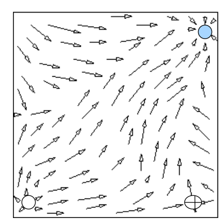


Critical points in GVF

- Critical points where gradient vanishes

$$\nabla f(p) = \mathbf{0}$$





Critical points in GVF - 1D

- Critical points where gradient vanishes

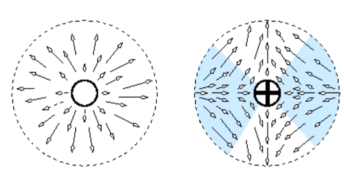
$$\nabla f(p) = \mathbf{0}$$

Critical points on a 1d domain

Critical points in GVF - 2D

- Critical points where gradient vanishes

$$\nabla f(p) = \mathbf{0}$$

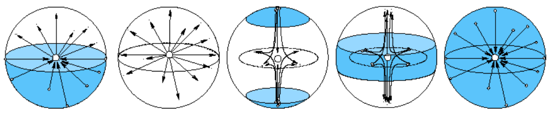


Critical points on a 2d domain

Critical points in GVF - 3D

- Critical points where gradient vanishes

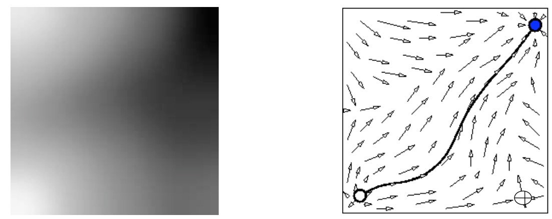
$$\nabla f(p) = \mathbf{0}$$



Critical points on a 3d domain

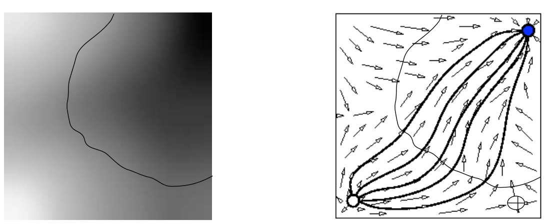
Integral Lines

- Integral Lines
  - Agrees with the gradient at every point on line
  - Has an *origin* and *destination* (lower and upper limits) that are critical points
    - These form boundaries where gradient is zero



Properties of Integral Lines

- Integral Lines
  - DO NOT CROSS
  - Are perpendicular to contours everywhere
  - Cover all non-critical points in entire domain



Ascending/Descending Manifolds

- Ascending/Descending Manifolds
  - Associated with a critical point *p*

Ascending / unstable	Descending / stable
The collection of points in integral lines where *p* is the origin	The collection of points in integral lines where *p* is the destination

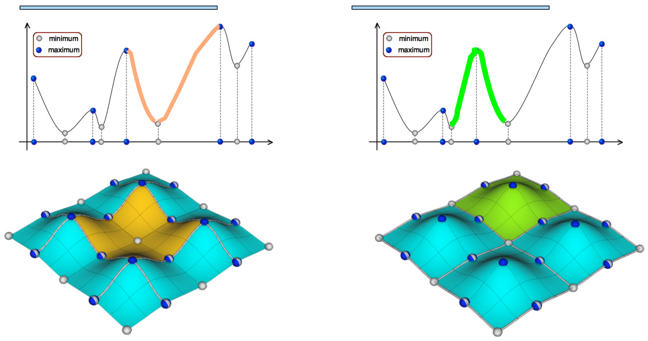
Ascending manifolds of *f* are descending manifolds of *-f*
Ascending manifolds form a cell complex
- boundary of ascending manifolds are lower dim asc manifolds
- two ascending manifolds do not intersect
- ascending manifolds partition *M*

Descending manifolds form a dual cell complex

Ascending/Descending Manifolds

- Ascending/Descending Manifolds

Ascending / unstable — Descending / stable



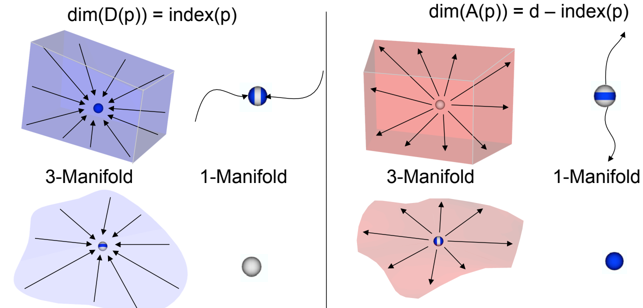
Ascending/Descending Manifolds

Descending Manifolds
$D(p) = \{p\} \cup \{ x \mid x \in l, \text{dest}(l) = p\}$

dim(D(p)) = index(p)

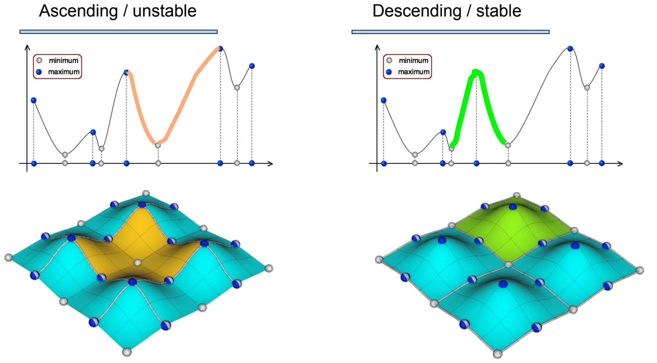
Ascending Manifolds
$A(p) = \{p\} \cup \{ x \mid x \in l, \text{orig}(l) = p\}$

dim(A(p)) = d – index(p)



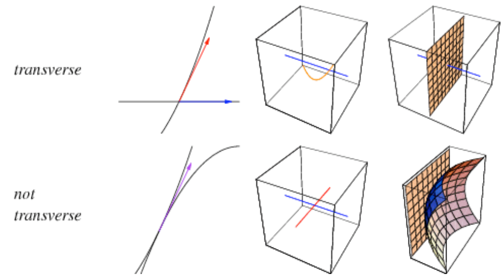
Morse Complex

- Ascending/Descending Morse Complex
  - Cell complex – partition of space



Morse-Smale function

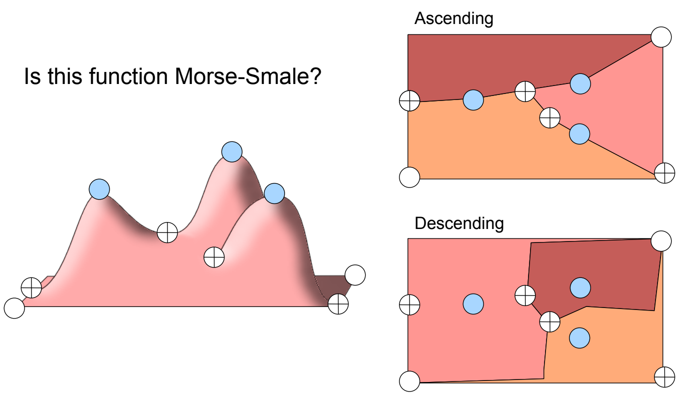
- A *Morse-Smale function* is a Morse function where ascending and descending manifolds intersect only transversally



Images from wikipedia

Morse Complex

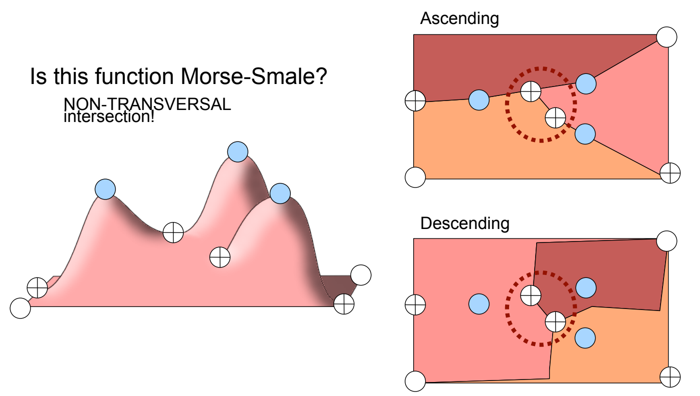
Is this function Morse-Smale?



Morse Complex

Is this function Morse-Smale?

NON-TRANSVERSAL intersection!



Morse-Smale function

- Transversal intersections:
  - Practically we say that
    - In 2d ascending/descending 1-manifolds cross at a point
    - In 3d ascending/descending 2-manifolds cross along arcs

- Can perturb a Morse function to make it Morse-Smale

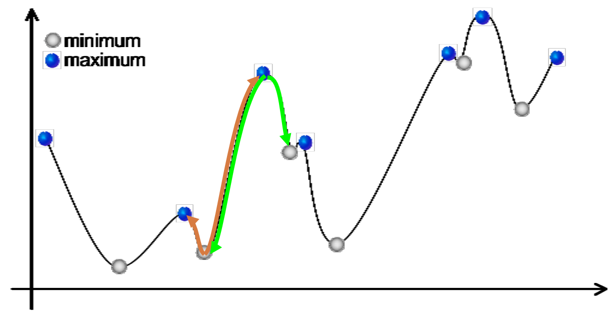
Morse-Smale complex

- Given a Morse-Smale function
- The cell complex formed by the intersection of the ascending and descending manifolds is the *Morse-Smale complex*

Morse-Smale complex

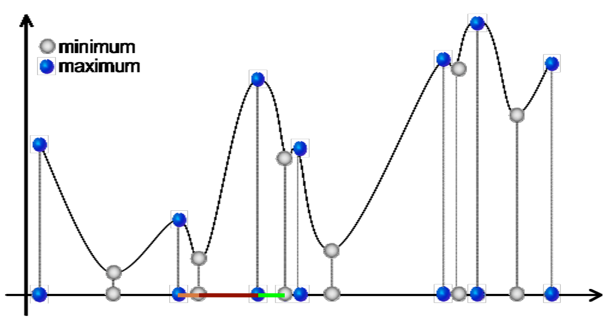
1D domain

Morse-Smale complex

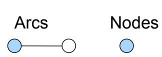


1D domain

Morse-Smale complex

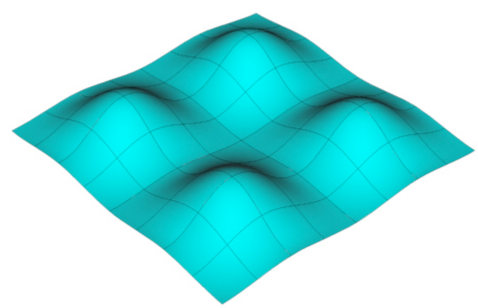


1D domain



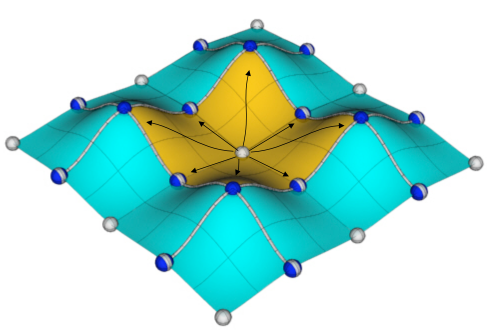
Morse-Smale complex

2D domain



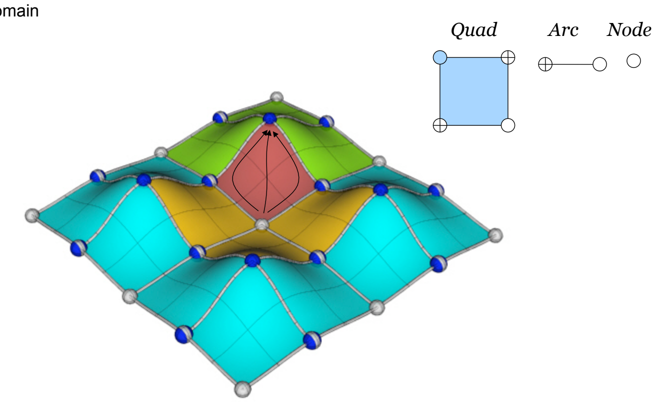
Morse-Smale complex

2D domain

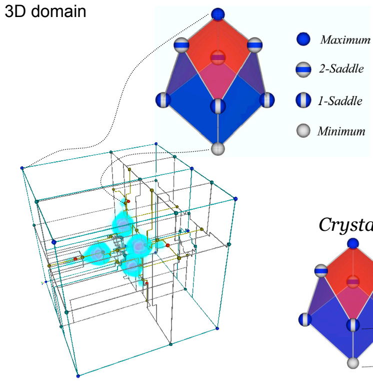


Morse-Smale complex

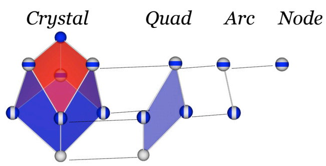
2D domain



Morse-Smale complex



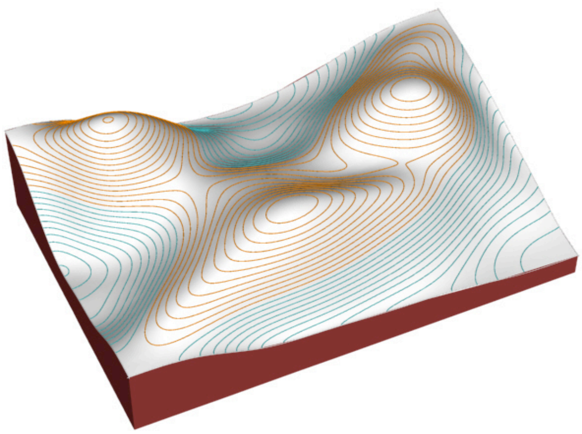
Cells of dimension i connect critical points with index that differ by i.



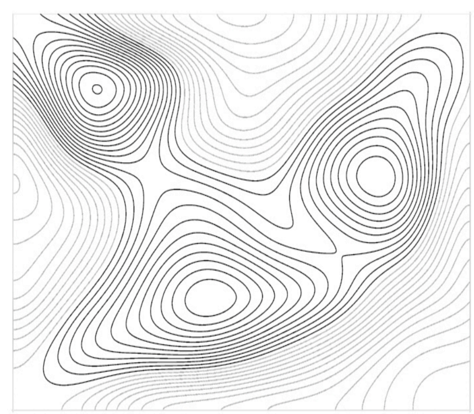
Morse-Smale complex

- Properties of cells
  - What do contours look like?
  - Can you extract the Morse complexes?
  - Can you extract the Contour Trees?
  - What is wrong with the definition:
    - A cell of the Morse-Smale complex is composed of the integral lines that share a common origin and destination?

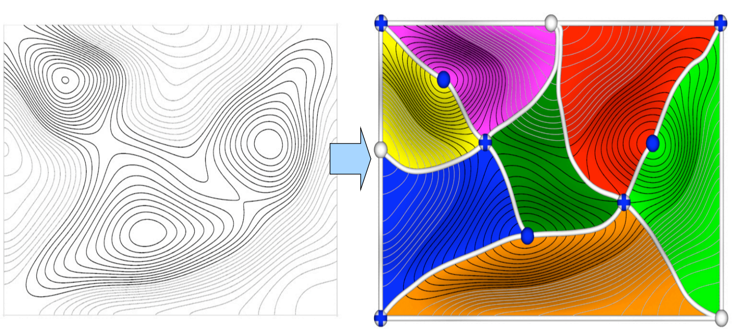
A Simple Example



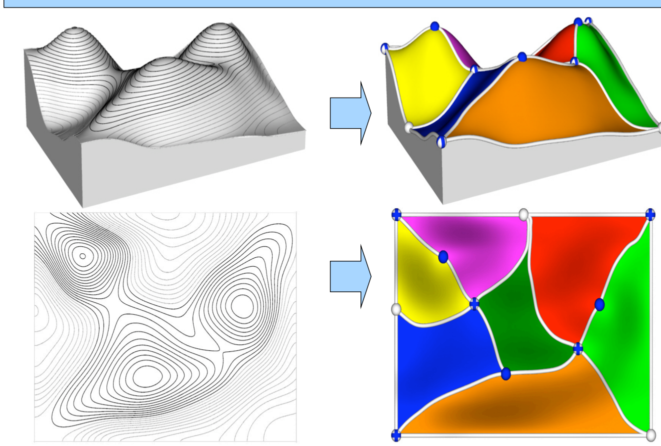
A Simple Example



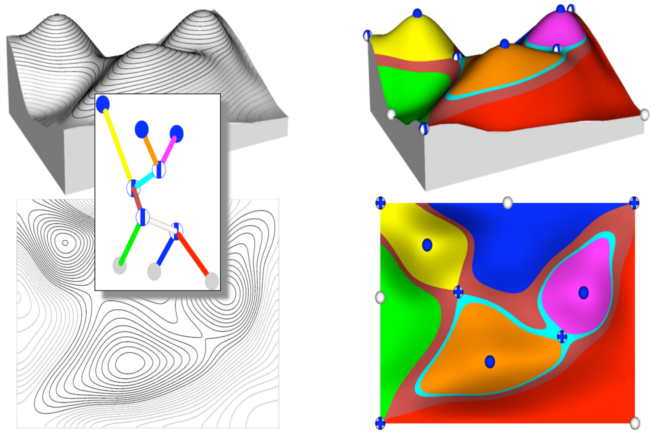
A Simple Example



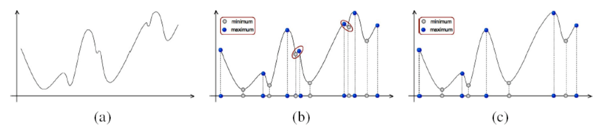
A Simple Example



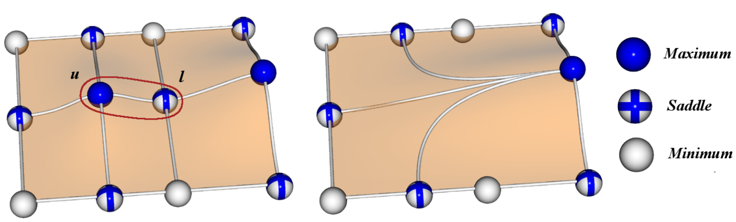
A Simple Example



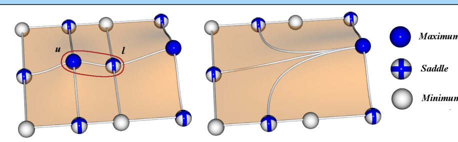
Simplification



Cancellations

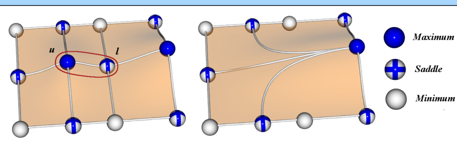


Cancellations – how does it look?



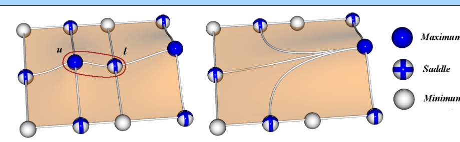
- What happens to the complex?
  - Cells are extended
  - Cells merge
  - Arcs merge
  - Arcs are deleted
  - Nodes are removed

Cancellations – a general definition



- Combinatorial change
  - Lower neighbors of the upper node are connected to upper neighbors of the lower node
  - All the arcs connected to *u* and *l* are removed
  - *u* and *l* are removed

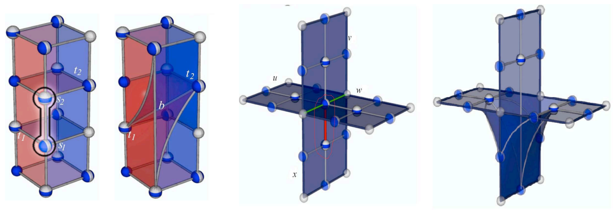
Cancellations – 2D case



- For the special case of 2D, a cancellation is the merging of an extremum-saddle-extremum

Cancellations – 3D case

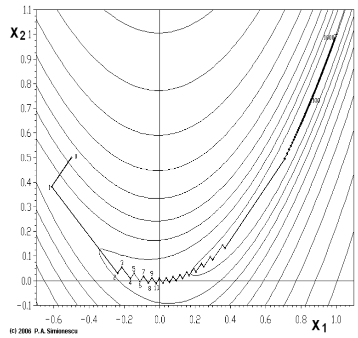
- Saddle-Extremum cancellations just as in 2D

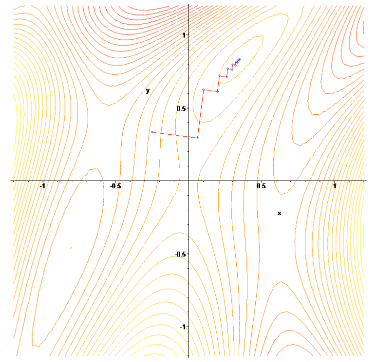


- 1Saddle-2Saddle cancellations

Computing the 2d MS complex

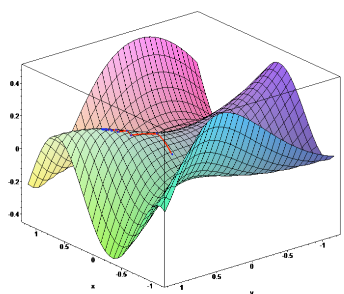
- Why don't we just integrate from every point?
  - Gradient descent





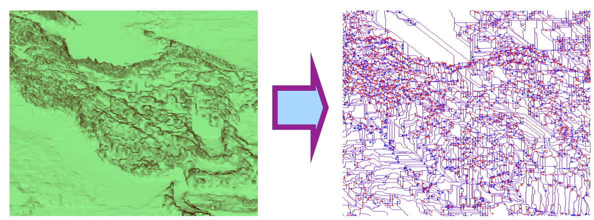
Computing the 2d MS complex

- Crossing integral lines!
  - Consistency – want quads
  - What do you do with degeneracies?
    - Flat regions
    - multi-saddles



Algorithm 1 – 2D MS complex

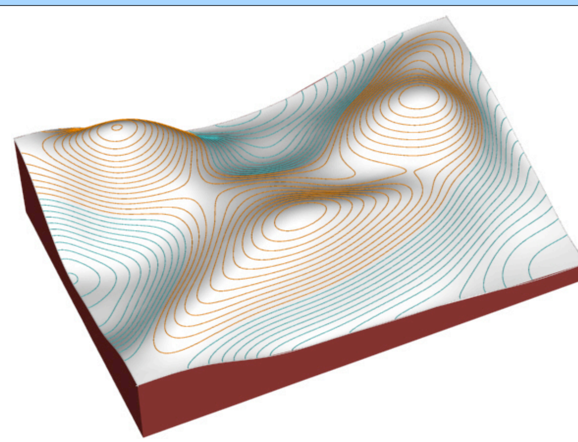
- Edelsbrunner et al. *Hierarchical Morse Complexes for Piecewise Linear 2-Manifolds*
  - Input: a scalar function on a simplicial 2-manifold mesh
    - Triangles, edges, vertices
  - Output: the Quasi-Morse-Smale complex



Algorithm 1 – 2D MS complex

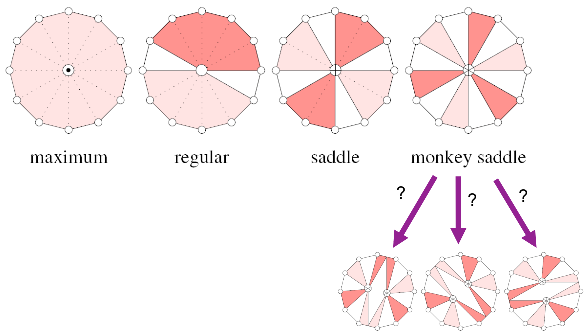
- 2-Stage Combinatorial algorithm
- (0) identify all critical points
- (1) trace descending paths from saddles
- (2) trace ascending paths from saddles, not allowing crosses of descending lines

A Simple Example

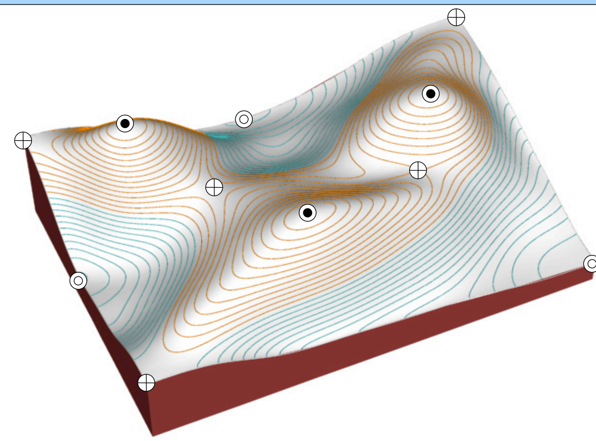


Algorithm 1 – 2D MS complex

- (0) identify all critical points

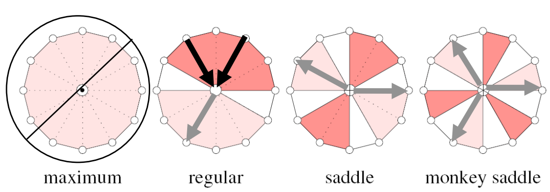


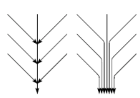
A Simple Example



Algorithm 1 – 2D MS complex

- (1) trace descending paths from saddles
  - Extend paths along steepest downward edge

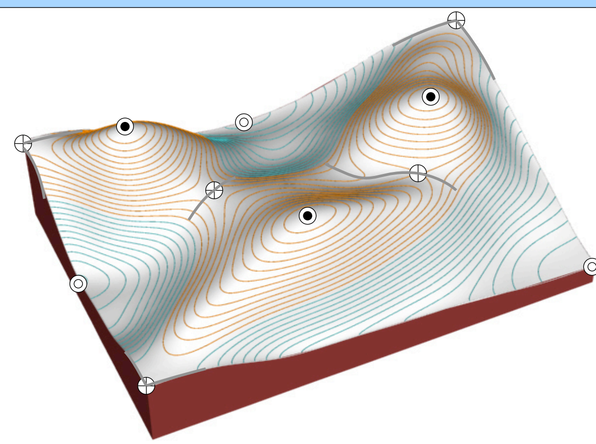




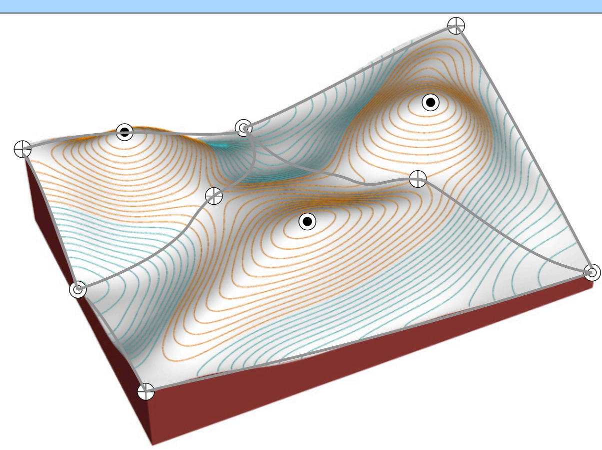
Merging paths are legal – represent a split between paths

ONCE THINGS MERGE THEY WILL NEVER SPLIT!!!!!!

A Simple Example

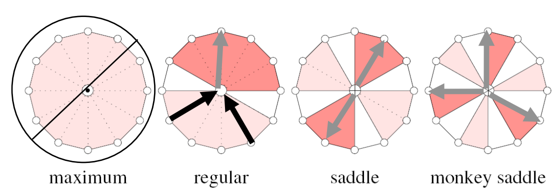


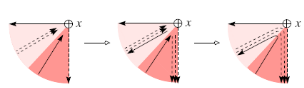
A Simple Example



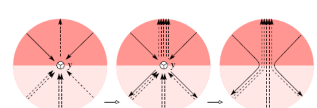
Algorithm 1 – 2D MS complex

- (2) trace ascending paths from saddles
  - Must be careful not to cross descending lines!!!!



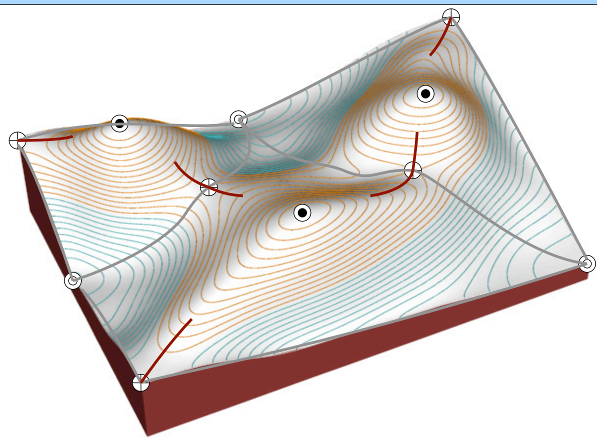


Paths ending at a saddle are forced around the saddle

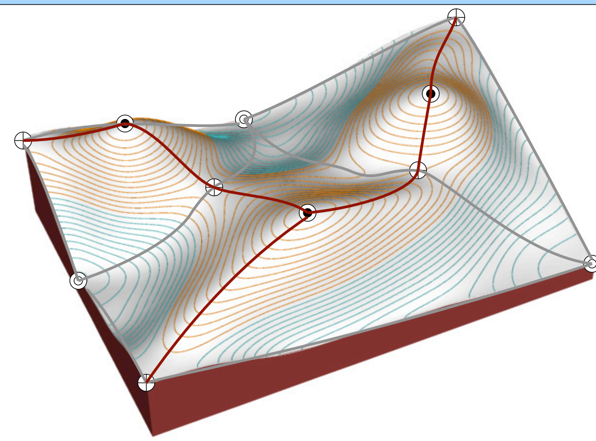


Ascending paths are forced not to intersect descending paths

A Simple Example

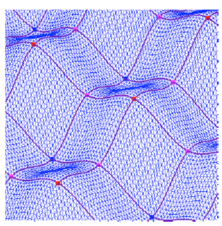


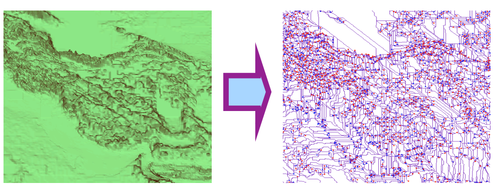
A Simple Example



Algorithm 1 – 2D MS complex

- Results
  - Combinatorial method
  - Handles degeneracies
  - Care has to be taken in keeping things separated





Algorithm 2 – 3D MS complex

- Why is simple extension of 2D ideas difficult?
  - Trace descending manifolds
    - Arcs and surfaces from saddles
  - Trace ascending manifolds, keeping them separate from existing descending manifolds

H. Edelsbrunner, J. Harer, V. Natarajan and V. Pascucci. Morse-Smale complexes for piecewise linear 3-manifolds. *in* ``Proc. 19th Ann. Sympos. Comput. Geom. 2003'', 361-370.

Algorithm 3 – Discrete MS complex

- Simple, generic, applicable to any dimension

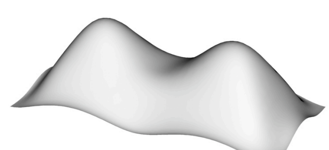
- Based on discrete Morse theory:

R. Forman. *A users guide to discrete Morse theory*. In Proc. of the 2001 Internat. Conf. on Formal Power Series and Algebraic Combinatorics, A special volume of Advances in Applied Mathematics, page 48, 2001.

T. Lewiner. *Constructing discrete Morse functions*. Master's thesis, Department of Mathematics, PUC-Rio, 2002.

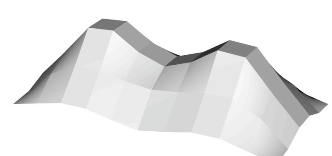
Continuous to discrete Morse theory

Continuous Case



Morse function

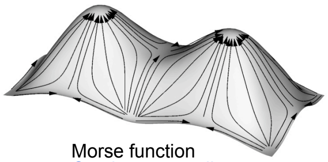
Discrete Case



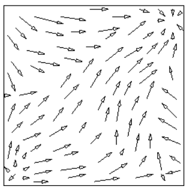
Discrete samples

Continuous to discrete Morse theory

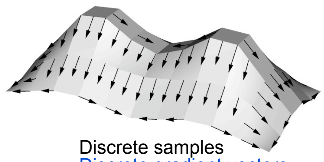
Continuous Case



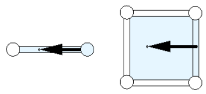
Morse function
Continuous gradient



Discrete Case

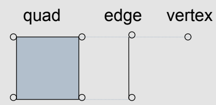


Discrete samples
Discrete gradient vectors

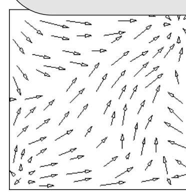


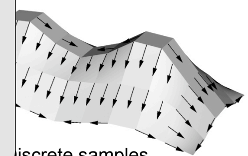
Pair a *(d-1)*-cell with a *d*-cell as the tail and head of a gradient arrow

Continuous to discrete Morse theory

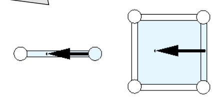


faces : cells in the closure of α
co-faces : cells whose closure contain α
facet : face of α with dimension different by one
co-facet : co-face of α with dimension different by one





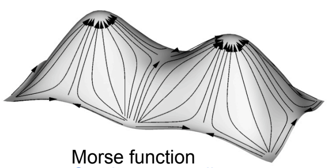
Discrete samples
Discrete gradient vectors



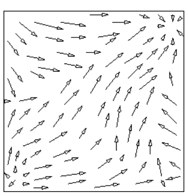
Pair a *(d-1)*-cell with a *d*-cell as the tail and head of a gradient arrow

Continuous to discrete Morse theory

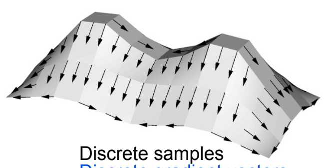
Continuous Case



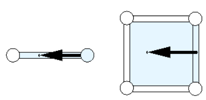
Morse function
Continuous gradient



Discrete Case



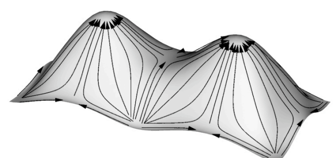
Discrete samples
Discrete gradient vectors



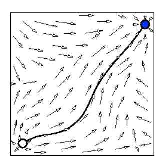
Pair a *(d-1)*-cell with a *d*-cell as the tail and head of a gradient arrow

Continuous to discrete Morse theory

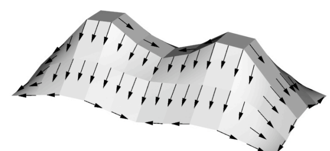
Continuous Case



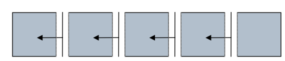
Morse function
Continuous gradient
Integral lines



Discrete Case



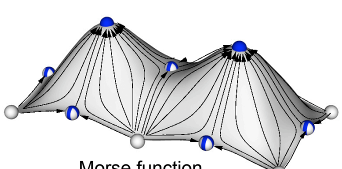
Discrete samples
Discrete gradient vectors
Gradient paths



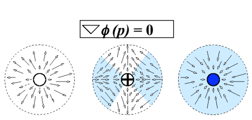
A sequence of gradient vectors

Continuous to discrete Morse theory

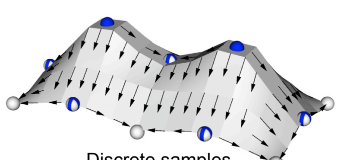
Continuous Case



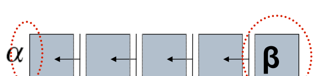
Morse function
Continuous gradient
Integral lines
Critical points



Discrete Case



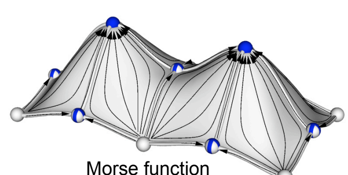
Discrete samples
Discrete gradient vectors
Gradient paths
Critical cells



Unpaired cells in the discrete gradient
Index = dimension of the cell

Continuous to discrete Morse theory

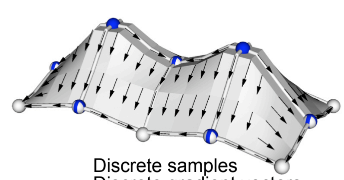
Continuous Case



Morse function
Continuous gradient
Integral lines
Critical points
Morse-Smale Complex

A cell consists of Integral lines sharing a common origin and destination

Discrete Case

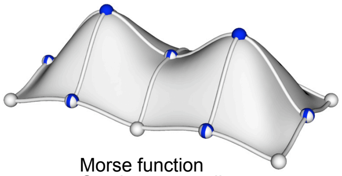


Discrete samples
Discrete gradient vectors
Gradient paths
Critical cells
Morse-Smale Complex

A cell consists of the sum of paths sharing a common origin and destination

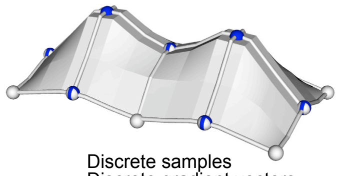
Continuous to discrete Morse theory

Continuous Case



Morse function
Continuous gradient
Integral lines
Critical points
Morse-Smale Complex

Discrete Case



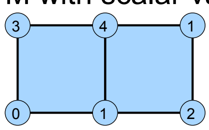
Discrete samples
Discrete gradient vectors
Gradient paths
Critical cells
Morse-Smale Complex

Algorithm 3 – Discrete MS complex

- Construct a discrete gradient vector field
- Trace ascending and descending manifolds

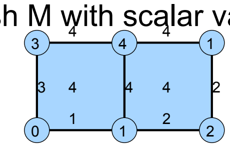
Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices



Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

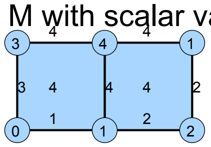


- Assign a value to every cell

$$F(\alpha) = MAX\{\sigma : \sigma < \alpha\} + \varepsilon$$

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

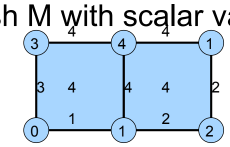


- Create gradient arrows in direction of steepest descent

For *i* = 0, …. , *d*
  While not all *i*-cells have been marked
    lowest = Lowest_Unmarked_iCell()
  If Can_Pair (*lowest*)
    Pair_With_Steepest_Descent_cofacet(*lowest*)
  Else
    Set_Critical(*lowest*)

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

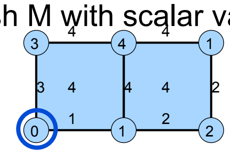


- Create gradient arrows in direction of steepest descent

For *i* = 0, …. , *d* ***i* = 0**
  While not all *i*-cells have been marked
    lowest = Lowest_Unmarked_iCell()
  If Can_Pair (*lowest*)
    Pair_With_Steepest_Descent_cofacet(*lowest*)
  Else
    Set_Critical(*lowest*)

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

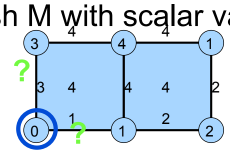


- Create gradient arrows in direction of steepest descent

```
For i = 0, …. , d   i = 0
    While not all i-cells have been marked
        lowest = Lowest_Unmarked_iCell()
    If Can_Pair (lowest)
        Pair_With_Steepest_Descent_cofacet(lowest)
    Else
        Set_Critical(lowest)
```

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

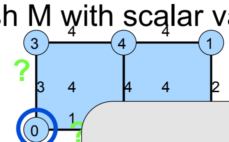


- Create gradient arrows in direction of steepest descent

```
For i = 0, …. , d   i = 0
    While not all i-cells have been marked
        lowest = Lowest_Unmarked_iCell()
    If Can_Pair (lowest)
        Pair_With_Steepest_Descent_cofacet(lowest)
    Else
        Set_Critical(lowest)
```

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices



- Create gradient arr… steepest descent

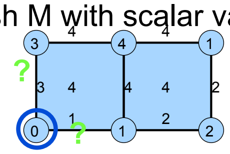
Can pair α with **β** if

α is only unmarked facet of **β** and $f(\alpha) < f(\boldsymbol{\beta})$

```
For i = 0, …. , d   i = 0
    While not all i-cells h…
        lowest = Lowest…
    If Can_Pair (lowest)
        Pair_With_Steepest_Descent_cofacet(lowest)
    Else
        Set_Critical(lowest)
```

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

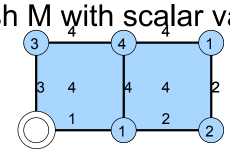


- Create gradient arrows in direction of steepest descent

```
For i = 0, …. , d   i = 0
    While not all i-cells have been marked
        lowest = Lowest_Unmarked_iCell()
    If Can_Pair (lowest)
        Pair_With_Steepest_Descent_cofacet(lowest)
    Else
        Set_Critical(lowest)
```

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

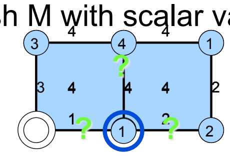


- Create gradient arrows in direction of steepest descent

```
For i = 0, …. , d   i = 0
    While not all i-cells have been marked
        lowest = Lowest_Unmarked_iCell()
    If Can_Pair (lowest)
        Pair_With_Steepest_Descent_cofacet(lowest)
    Else
        Set_Critical(lowest)
```

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

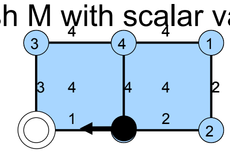


- Create gradient arrows in direction of steepest descent

```
For i = 0, …. , d   i = 0
    While not all i-cells have been marked
        lowest = Lowest_Unmarked_iCell()
    If Can_Pair (lowest)
        Pair_With_Steepest_Descent_cofacet(lowest)
    Else
        Set_Critical(lowest)
```

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

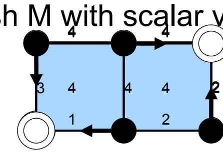


- Create gradient arrows in direction of steepest descent

For *i* = 0, …. , *d* ***i* = 0**
While not all *i*-cells have been marked
lowest = Lowest_Unmarked_iCell()
If Can_Pair (*lowest*)
Pair_With_Steepest_Descent_cofacet(*lowest*)
Else
Set_Critical(*lowest*)

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

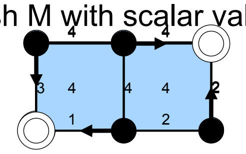


- Create gradient arrows in direction of steepest descent

For *i* = 0, …. , *d* ***i* = 0**
While not all *i*-cells have been marked
lowest = Lowest_Unmarked_iCell()
If Can_Pair (*lowest*)
Pair_With_Steepest_Descent_cofacet(*lowest*)
Else
Set_Critical(*lowest*)

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

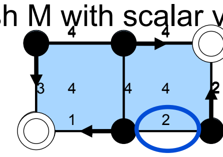


- Create gradient arrows in direction of steepest descent

For *i* = 0, …. , *d* ***i* = 1**
While not all *i*-cells have been marked
lowest = Lowest_Unmarked_iCell()
If Can_Pair (*lowest*)
Pair_With_Steepest_Descent_cofacet(*lowest*)
Else
Set_Critical(*lowest*)

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

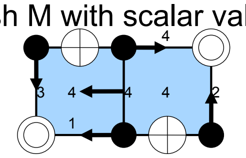


- Create gradient arrows in direction of steepest descent

For *i* = 0, …. , *d* ***i* = 1**
While not all *i*-cells have been marked
lowest = Lowest_Unmarked_iCell()
If Can_Pair (*lowest*)
Pair_With_Steepest_Descent_cofacet(*lowest*)
Else
Set_Critical(*lowest*)

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

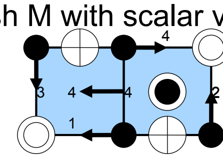


- Create gradient arrows in direction of steepest descent

For *i* = 0, …. , *d* ***i* = 1**
While not all *i*-cells have been marked
lowest = Lowest_Unmarked_iCell()
If Can_Pair (*lowest*)
Pair_With_Steepest_Descent_cofacet(*lowest*)
Else
Set_Critical(*lowest*)

Computing the discrete gradient on a mesh

- Given a mesh M with scalar valued defined at vertices

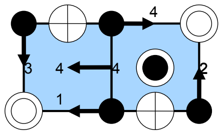


- Create gradient arrows in direction of steepest descent

For *i* = 0, …. , *d* ***i* = 2**
While not all *i*-cells have been marked
lowest = Lowest_Unmarked_iCell()
If Can_Pair (*lowest*)
Pair_With_Steepest_Descent_cofacet(*lowest*)
Else
Set_Critical(*lowest*)

Computing the discrete gradient on a mesh

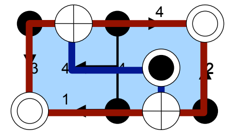
- Now we have a discrete gradient!



- The Morse-Smale complex is given by simply tracing gradient paths
  - Arcs are paths that start and end at critical cells

Computing the discrete gradient on a mesh

- Now we have a discrete gradient!



- The Morse-Smale complex is given by simply tracing gradient paths
  - Arcs are paths that start and end at critical cells

References

- 2d Morse-Smale complexes

  H. Edelsbrunner, J. Harer and A. Zomorodian. *Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds.* Discrete Comput. Geom. 30 (2003), 87-107.

  P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci *A topological hierarchy for functions on triangulated surfaces.* IEEE Trans. on Visualization and Computer Graphics 2004

- 3d Morse-Smale complexes

  H. Edelsbrunner, J. Harer, V. Natarajan and V. Pascucci. *Morse-Smale complexes for piecewise linear 3-manifolds.* In ``Proc. 19th Ann. Sympos. Comput. Geom. 2003'', 361-370.

  Attila Gyulassy, Vijay Natarajan, Valerio Pascucci, Peer-Timo Bremer, Bernd Hamann *Topology-based Simplification for Feature Extraction from 3D Scalar Fields.* IEEE Visualization (IEEE Visualization 2005), pp 535-542, 2005.

  Attila Gyulassy, Peer-Timo Bremer, Valerio Pascucci, Bernd Hamann. *A Practical Approach to Morse Smale Complex Computation: Scalability and Generality.* IEEE Trans. Vis. Comput. Graph. (IEEE Visualization 2008), 14(6): 1619-1626, 2008.

Questions?